**ЛАБОРАТОРНА РОБОТА №6**

**КОНСТРУКЦІЯ SELECT**

Мета: основною метою лабораторної роботи є навчити студентів правильно застосовувати та розуміти конструкцію SELECT в SQL, яка є фундаментальним інструментом для вибірки даних з реляційних баз даних.

Цілі лабораторної роботи включають:

1. Освоєння базової структури: студенти навчаться формувати базові запити SELECT, що включають вибірку колонок з однієї таблиці.

2. Розуміння Клауз WHERE: здобуття навичок використання клаузи WHERE для фільтрації даних згідно з певними умовами.

3. Використання агрегатних функцій: освоєння застосування агрегатних функцій, таких як COUNT, SUM, AVG, MIN, та MAX для аналізу даних.

4. Застосування Клауз GROUP BY і HAVING: вивчення групування даних за допомогою GROUP BY та фільтрація групованих результатів за допомогою HAVING.

5. Освоєння порядку сортування даних: розуміння та застосування клаузи ORDER BY для сортування результатів запиту.

7. Оптимізація вибірки: введення в практику використання DISTINCT, TOP, і OFFSET FETCH для оптимізації вибірки даних та управління обсягами результатів.

Ця лабораторна робота сприятиме зміцненню знань студентів у сфері роботи з базами даних та підготує їх до вирішення складних задач аналізу та обробки даних.

**ТЕОРЕТИЧНІ ОСНОВИ**

**Компоненти конструкції SELECT**

|  |  |  |  |
| --- | --- | --- | --- |
| Порядок написання у запиті | Компоненти конструкції SELECT | Що виконує компонента | Порядок виконання СУБД |
| **1** | SELECT | Визначає найменування стовбців для виводу даних | **5** |
| **2** | FROM | Визначає найменування таблиць | **1** |
| **3** | WHERE | Здійснює пошук рядків за якимось критерієм | **2** |
| **4** | GROUP BY | Формує групи рядків | **3** |
| **5** | HAVING | Фільтрує групи рядків | **4** |
| **6** | ORDER BY | Здійснює сортування результуючого набору даних | **6** |

**Основна форма запиту SELECT виглядає так:**

SELECT column1, column2, ...

FROM table\_name;

*Приклад:*

SELECT FirstName, LastName -- З таблиці відбираються тільки два стовбця

FROM Employees;

------------------------------------------------------------------

SELECT \* -- З таблиці відбираються всі стовбці

FROM Employees;

**Розуміння Клаузи WHERE**

Клауза WHERE використовується для фільтрації рядків за певними критеріями.

*Приклад:*

SELECT FirstName, LastName, Age

FROM Employees

WHERE Age > 30; -- Відбираються рядки, де вік співробітника > 30

------------------------------

WHERE HIRE\_DATE BETWEEN '20040101' AND '20041231'

-- Відбирається рядки, де дата найму знаходиться в зазначеному проміжку

**Використання Агрегатних Функцій**

Агрегатні функції використовуються для виконання обчислень на наборі значень та повернення одного значення.

**Приклад**

SELECT COUNT(\*) AS [Кількість рядків]

FROM [dbo].[SALESREPS];

SELECT SUM(SALES) AS [Загальна сума поточних продажів]

FROM [dbo].[SALESREPS];

SELECT MIN(AGE) AS [Найменший вік співробітників]

FROM [dbo].[SALESREPS];

SELECT MAX(AGE) AS [Найбільший вік співробітників]

FROM [dbo].[SALESREPS];

SELECT AVG(AGE) AS [Середній вік співробітників]

FROM [dbo].[SALESREPS]

**Застосування Клауз GROUP BY і HAVING**

GROUP BY використовується для групування рядків, що мають однакові значення в певній колонці, а HAVING для фільтрації цих груп.

**Приклад:**

SELECT DepartmentID, COUNT(EmployeeID)

FROM Employees

GROUP BY DepartmentID

HAVING COUNT(EmployeeID) > 10;

-------------------------------------------------------

SELECT [TITLE]

,[MANAGER]

,COUNT (\*)

FROM [dbo].[SALESREPS]

GROUP BY [TITLE],[MANAGER];

**Порядок Сортування Даних**

ORDER BY використовується для сортування рядків у вибірці за однією або декількома колонками.

**Приклад:**

SELECT [TITLE]

,COUNT(DISTINCT EMPL\_NUM) AS [Qty]

FROM [dbo].[SALESREPS]

WHERE HIRE\_DATE BETWEEN '20040101' AND '20041231'

GROUP BY [TITLE]

ORDER BY [Qty] DESC --DESC - сортування за зменшенням // ASC - сортування за збільшенням

;

**Оптимізація Вибірки**

Для оптимізації вибірки даних можна використовувати DISTINCT, TOP, WITH TIES, OFFSET FETCH.

**DISTINCT**

Оператор `DISTINCT` використовується для видалення дублікатів з результатів запиту. Якщо ви вибираєте декілька колонок, `DISTINCT` застосовується до їх комбінації.

**Приклад:**

SELECT DISTINCT FirstName, LastName

FROM Employees;

Цей запит вибере унікальні комбінації імен та прізвищ з таблиці `Employees`.

**TOP**

Клауза TOP використовується для обмеження кількості рядків, які повертаються запитом. Вона може бути корисною, наприклад, коли потрібно отримати тільки перші N записів результату.

**Приклад:**

SELECT TOP 10 \*

FROM Employees

ORDER BY Salary DESC;

Цей запит вибере 10 працівників з найвищою зарплатою.

**WITH TIES**

Клауза `WITH TIES` використовується з `TOP`, щоб повернути додаткові рядки, які співпадають з останнім рядком в результаті. Це корисно, коли ви хочете включити всі рядки, які мають таке ж значення певної колонки, як і останній рядок у TOP N.

**Приклад:**

SELECT TOP 10 WITH TIES \*

FROM Employees

ORDER BY Salary DESC;

Цей запит вибере працівників з найвищою зарплатою, але якщо на 10-му місці буде кілька працівників з однаковою зарплатою, він включить їх усіх.

**OFFSET FETCH**

`OFFSET FETCH` частина конструкції `ORDER BY` дозволяє пропустити задану кількість рядків (OFFSET) і потім повернути задану кількість рядків (FETCH). Це часто використовується для реалізації пагінації в запитах.

Приклад:

SELECT \*

FROM Employees

ORDER BY EmployeeID

OFFSET 20 ROWS FETCH NEXT 10 ROWS ONLY;

Цей запит пропустить перші 20 рядків (співробітників за ідентифікатором) і поверне наступні 10, що ефективно дає вам "другу сторінку" результатів, якщо ви вважаєте кожні 10 рядків "сторінкою".

**ЗАВДАННЯ**

**Завдання 1**

Напишіть запит, який поверне список працівників старше 45 років.

- Використовується таблиця [dbo].[SALESREPS]

- Результуючий набір даних містить:

ідентифікатор працівника, ім'я працівника, вік, посаду

- Відсортуйте результуючий набір даних за віком (за спаданням)

**Завдання 2**

Напишіть запит, який поверне список унікальних комбінацій значень

ідентифікатора виробника (MFR) та ідентифікатора товару (PRODUCT). Враховуйте тільки

замовлення за 2008 рік.

- Використовується таблиця [dbo].[ORDERS]

- Задійте оператор DISTINCT

- Результуючий набір даних містить: ідентифікатор виробника, ідентифікатор товару

**Завдання 3**

Напишіть запит, який поверне ідентифікатор працівника ([REP]) з найбільшою кількістю проведених замовлень.

Враховуйте можливість того, що одразу кілька працівників можуть мати однакову кількість проведених замовлень.

Враховуйте тільки замовлення за 2008 рік.

- Використовується таблиця [dbo].[ORDERS]

- Задійте агрегатну функцію COUNT

- Задійте оператор WITH TIES

- Результуючий набір даних містить: Ідентифікатор працівника,

кількість проведених замовлень

**КОНТРОЛЬНІ ЗАПИТАННЯ**

1. Яку основну мету виконує оператор SELECT у SQL?

2. Як використовується клауза DISTINCT і для чого вона служить?

3. Як можна обмежити кількість рядків, що повертаються запитом SELECT?

4. Що робить клауза WHERE і як вона використовується у запиті SELECT?

5. Як застосовується клауза ORDER BY та які опції сортування вона дозволяє виконати?

6. Що таке агрегатні функції і назвіть кілька прикладів таких функцій.

7. Як клауза GROUP BY відрізняється від ORDER BY і коли її слід використовувати?

8. Що робить клауза HAVING і як вона відрізняється від WHERE?

9. Як можна використовувати клаузу TOP разом із WITH TIES?

10. Як працює пагінація даних за допомогою OFFSET FETCH?

11. Як ви можете вибрати унікальні записи з таблиці без повторень?

12. Що робити, якщо вам потрібно вибрати дані, базуючись на умовах, пов'язаних із значеннями іншої таблиці?

13. Як ви можете змінити порядок сортування результатів запиту SELECT?

14. Які можливості для оптимізації вибірки даних пропонує T-SQL, і як їх можна використовувати у складних запитах?

**ДОДАТКОВІ МАТЕРІАЛИ:**

https://www.w3schools.com/sql/sql\_select.asp